Neurocomputing 160 (2015) 53–62

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Support vector machines under adversarial label contamination

Huang Xiao [a], Battista Biggio [b,*], Blaine Nelson [b], Han Xiao [a], Claudia Eckert [a], Fabio Roli [b]

[a] Department of Computer Science, Technical University of Munich, Boltzmannstr. 3, 85748 Garching, Germany
[b] Department of Electrical and Electronic Engineering, University of Cagliari, Piazza d'Armi, 09123 Cagliari, Italy

ARTICLE INFO

Article history:
Received 15 March 2014
Received in revised form
26 July 2014
Accepted 1 August 2014
Available online 10 February 2015

Keywords:
Support vector machines
Adversarial learning
Label noise
Label flip attacks

ABSTRACT

Machine learning algorithms are increasingly being applied in security-related tasks such as spam and malware detection, although their security properties against deliberate attacks have not yet been widely understood. Intelligent and adaptive attackers may indeed exploit specific vulnerabilities exposed by machine learning techniques to violate system security. Being robust to adversarial data manipulation is thus an important, additional requirement for machine learning algorithms to successfully operate in adversarial settings. In this work, we evaluate the security of Support Vector Machines (SVMs) to well-crafted, adversarial label noise attacks. In particular, we consider an attacker that aims to maximize the SVM's classification error by flipping a number of labels in the training data. We formalize a corresponding optimal attack strategy, and solve it by means of heuristic approaches to keep the computational complexity tractable. We report an extensive experimental analysis on the effectiveness of the considered attacks against linear and non-linear SVMs, both on synthetic and real-world datasets. We finally argue that our approach can also provide useful insights for developing more secure SVM learning algorithms, and also novel techniques in a number of related research areas, such as semi-supervised and active learning.

1. Introduction

Machine learning and pattern recognition techniques are increasingly being adopted in security applications like spam, intrusion and malware detection, despite their security to adversarial attacks has not yet been deeply understood. In adversarial settings, indeed, intelligent and adaptive attackers may carefully target the machine learning components of a system to compromise its security. Several distinct attack scenarios have been considered in a recent field of study, known as *adversarial machine learning* [1–4]. For instance, it has been shown that it is possible to gradually *poison* a spam filter, an intrusion detection system, and even a biometric verification system (in general, a classification algorithm) by exploiting update mechanisms that enable the adversary to manipulate some of the training data [5–13]; and that the detection of malicious samples by linear and even some classes of non-linear classifiers can be *evaded* with few targeted manipulations that reflect a proper change in their feature values [14,13,15–17]. Recently, poisoning and evasion attacks against clustering algorithms have also been formalized to show that malware clustering approaches can be significantly vulnerable to well-crafted attacks [18,19].

Research in adversarial learning not only investigates the security properties of learning algorithms against well-crafted attacks, but it also focuses on the development of more secure learning algorithms. For evasion attacks, this has been mainly achieved by explicitly embedding knowledge into the learning algorithm of the possible data manipulation that can be performed by the attacker, *e.g.*, using game-theoretical models for classification [15,20–22], probabilistic models of the data distribution drift under attack [23,24], and even multiple classifier systems [25–27]. Poisoning attacks and manipulation of the training data have been differently countered with data sanitization (*i.e.*, a form of outlier detection) [5,6,28], multiple classifier systems [29], and robust statistics [7]. Robust statistics have also been exploited to formally show that the *influence function* of SVM-like algorithms can be bounded under certain conditions [30]; *e.g.*, if the kernel is bounded. This ensures some degree of robustness against small perturbations of training data, and it may be thus desirable also to improve the security of learning algorithms against poisoning.

In this work, we investigate the vulnerability of SVMs to a specific kind of training data manipulation, *i.e.*, worst-case label noise. This can be regarded as a carefully crafted attack in which the labels of a subset of the training data are flipped to maximize the SVM's classification error. While stochastic label noise has been widely studied in the machine learning literature, to account for different kinds of potential labeling errors in the training data [31,32], only a few works have considered adversarial, worst-case label noise, either from a more theoretical [33] or practical perspective [34,35]. In

* Corresponding author
E-mail addresses: xiaohu@in.tum.de (H. Xiao), battista.biggio@diee.unica.it (B. Biggio), blaine.nelson@gmail.com (B. Nelson), xiaoh@in.tum.de (H. Xiao), claudia.eckert@in.tum.de (C. Eckert), roli@diee.unica.it (F. Roli).

http://dx.doi.org/10.1016/j.neucom.2014.08.081
0925-2312/© 2015 Elsevier B.V. All rights reserved.

[31,33] the impact of stochastic and adversarial label noise on the classification error have been theoretically analyzed under the *probably approximately correct* learning model, deriving lower bounds on the classification error as a function of the fraction of flipped labels η; in particular, the test error can be shown to be lower bounded by $\eta/(1-\eta)$ and 2η for stochastic and adversarial label noise, respectively. In recent work [34,35], instead, we have focused on deriving more practical attack strategies to maximize the test error of an SVM given a maximum number of allowed label flips in the training data. Since finding the worst label flips is generally computationally demanding, we have devised suitable heuristics to find approximate solutions efficiently. To our knowledge, these are the only works devoted to understanding how SVMs can be affected by adversarial label noise.

From a more practical viewpoint, the problem is of interest as attackers may concretely have access and change some of the training labels in a number of cases. For instance, if feedback from end-users is exploited to label data and update the system, as in collaborative spam filtering, an attacker may have access to an authorized account (*e.g.,* an email account protected by the same anti-spam filter), and manipulate the labels assigned to her samples. In other cases, a system may even ask directly to users to validate its decisions on some submitted samples, and use them to update the classifier (see, *e.g.,* PDFRate,[1] an online tool for detecting PDF malware [36]). The practical relevance of poisoning attacks has also been recently discussed in the context of the detection of malicious crowdsourcing websites that connect paying users with workers willing to carry out malicious campaigns (*e.g.,* spam campaigns in social networks) — a recent phenomenon referred to as *crowdturfing*. In fact, administrators of crowdturfing sites can intentionally pollute the training data used to learn classifiers, as it comes from their websites, thus being able to launch poisoning attacks [37].

In this paper, we extend our work on adversarial label noise against SVMs [34,35] by improving our previously defined attacks (Sections 3.1 and 3.3), and by proposing two novel heuristic approaches. One has been inspired from previous work on SVM poisoning [12] and incremental learning [38,39], and makes use of a continuous relaxation of the label values to greedily maximize the SVM's test error through gradient ascent (Section 3.2). The other exploits a breadth first search to greedily construct sets of candidate label flips that are *correlated* in their effect on the test error (Section 3.4). As in [34,35], we aim at analyzing the maximum performance degradation incurred by an SVM under adversarial label noise, to assess whether these attacks can be considered a relevant threat. We thus assume that the attacker has perfect knowledge of the attacked system and of the training data, and left the investigation on how to develop such attacks having limited knowledge of the training data to future work. We further assume that the adversary incurs the same cost for flipping each label, independently from the corresponding data point. We demonstrate the effectiveness of the proposed approaches by reporting experiments on synthetic and real-world datasets (Section 4). We conclude in Section 5 with a discussion on the contributions of our work, its limitations, and future research, also related to the application of the proposed techniques to other fields, including semi-supervised and active learning.

2. Support vector machines and notation

We revisit here structural risk minimization and SVM learning, and introduce the framework that will be used to motivate our attack strategies for adversarial label noise.

In risk minimization, the goal is to find a hypothesis $f: \mathcal{X} \rightarrow \mathcal{Y}$ that represents an unknown relationship between an input $\mathcal{X}$ and an output space $\mathcal{Y}$, captured by a probability measure P. Given a non-negative *loss function* $\ell : \mathcal{Y} \times \mathcal{Y} \rightarrow \mathbb{R}$ assessing the error between the prediction $\hat{y}$ provided by f and the true output y, we can define the optimal hypothesis $f^\star$ as the one that minimizes the expected risk $R(f,P) = \mathbb{E}_{(\mathbf{x},y)\sim P}[\ell(f(\mathbf{x}),y)]$ over the hypothesis space $\mathcal{F}$, *i.e.,* $f^\star = \arg\min_{f\in\mathcal{F}} R(f,P)$. Although P is not usually known, and thus $f^\star$ cannot be computed directly, a set $\mathcal{D}_{\text{tr}} = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$ of i.i.d. samples drawn from P are often available. In these cases a learning algorithm $\mathfrak{L}$ can be used to find a suitable hypothesis. According to structural risk minimization [40], the learner $\mathfrak{L}$ minimizes a sum of a regularizer and the empirical risk over the data:

$$\mathfrak{L}(\mathcal{D}_{\text{tr}}) = \arg\min_{f\in\mathcal{F}} \left[\Omega(f) + C\cdot \hat{R}(f,\mathcal{D}_{\text{tr}})\right], \tag{1}$$

where the regularizer $\Omega(f)$ is used to penalize excessive hypothesis complexity and avoid overfitting, the empirical risk $\hat{R}(f,\mathcal{D}_{\text{tr}})$ is given by $(1/n)\sum_{i=1}^n \ell(f(\mathbf{x}_i), y_i)$, and $C>0$ is a parameter that controls the trade-off between minimizing the empirical loss and the complexity of the hypothesis.

The SVM is an example of a binary linear classifier developed according to the aforementioned principle. It makes predictions in $\mathcal{Y} = \{-1,+1\}$ based on the sign of its real-valued discriminant function $f(\mathbf{x}) = \mathbf{w}^\top \mathbf{x} + b$; *i.e.,* $\mathbf{x}$ is classified as positive if $f(\mathbf{x}) \geq 0$, and negative otherwise. The SVM uses the hinge loss $\ell(f(\mathbf{x}),y) = \max(0, 1-yf(\mathbf{x}))$ as a convex surrogate loss function, and a quadratic regularizer on $\mathbf{w}$, *i.e.,* $\Omega(f) = \frac{1}{2}\mathbf{w}^\top\mathbf{w}$. Thus, SVM learning can be formulated according to Eq. (1) as the following convex quadratic programming problem:

$$\min_{\mathbf{w},b} \; \frac{1}{2}\mathbf{w}^\top\mathbf{w} + C\sum_{i=1}^n \max(0, 1-y_i f(\mathbf{x}_i)). \tag{2}$$

An interesting property of SVMs arises from their *dual* formulation, which only requires computing inner products between samples during training and classification, thus avoiding the need of an *explicit* feature representation. Accordingly, non-linear decision functions in the input space can be learned using *kernels*, *i.e.,* inner products in implicitly mapped feature spaces. In this case, the SVM's decision function is given as $f(\mathbf{x}) = \sum_{i=1}^n \alpha_i y_i k(\mathbf{x},\mathbf{x}_i) + b$, where $k(\mathbf{x},\mathbf{z}) = \phi(\mathbf{x})^\top\phi(\mathbf{z})$ is the kernel function, and ϕ the implicit mapping. The SVM's dual parameters $(\boldsymbol{\alpha}, b)$ are found by solving the dual problem:

$$\min_{0\leq\boldsymbol{\alpha}\leq C} \; \tfrac{1}{2}\boldsymbol{\alpha}^\top\mathbf{Q}\boldsymbol{\alpha} - \mathbf{1}^\top\boldsymbol{\alpha} \quad \text{s.t. } \mathbf{y}^\top\boldsymbol{\alpha} = 0, \tag{3}$$

where $\mathbf{Q} = \mathbf{y}\mathbf{y}^\top \circ \mathbf{K}$ is the label-annotated version of the (training) kernel matrix $\mathbf{K}$. The bias b is obtained from the corresponding Karush–Kuhn–Tucker (KKT) conditions, to satisfy the equality constraint $\mathbf{y}^\top\boldsymbol{\alpha} = 0$ (see, *e.g.,* [41]).

In this paper, however, we are not only interested in how the hypothesis is chosen but also how it performs on a second validation or test dataset $\mathcal{D}_{\text{vd}}$, which may be generally drawn from a different distribution Q. We thus define the error measure

$$V_{\mathfrak{L}}(\mathcal{D}_{\text{tr}}, \mathcal{D}_{\text{vd}}) = \|f_{\mathcal{D}_{\text{tr}}}\|^2 + C\cdot\hat{R}(f_{\mathcal{D}_{\text{tr}}}, \mathcal{D}_{\text{vd}}), \tag{4}$$

which implicitly uses $f_{\mathcal{D}_{\text{tr}}} = \mathfrak{L}(\mathcal{D}_{\text{tr}})$. This function evaluates the structural risk of a hypothesis $f_{\mathcal{D}_{\text{tr}}}$ that is *trained* on $\mathcal{D}_{\text{tr}}$ but *evaluated* on $\mathcal{D}_{\text{vd}}$, and will form the foundation for our label flipping approaches to dataset poisoning. Moreover, since we are only concerned with label flips and their effect on the learner we use the notation $V_{\mathfrak{L}}(\mathbf{z},\mathbf{y})$ to denote the above error measure when the datasets differ only in the labels $\mathbf{z}$ used for training and $\mathbf{y}$ used for evaluation; *i.e.,* $V_{\mathfrak{L}}(\mathbf{z},\mathbf{y}) = V_{\mathfrak{L}}(\{(\mathbf{x}_i,z_i)\}\{(\mathbf{x}_i,y_i)\})$.

[1] Available at: http://pdfrate.com